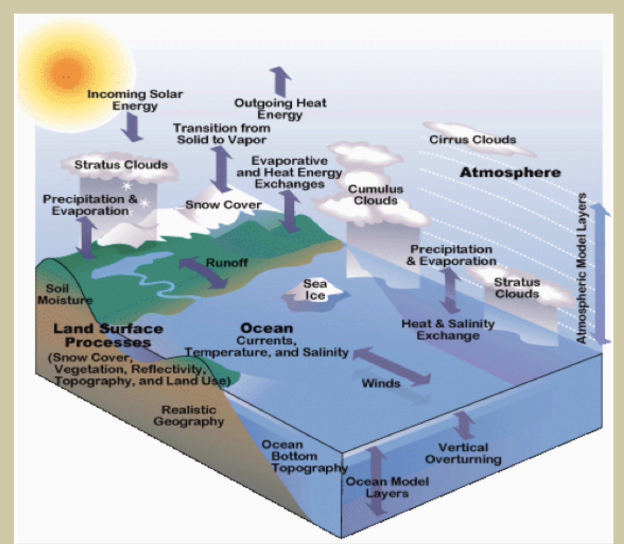
OAK RIDGE National Laboratory

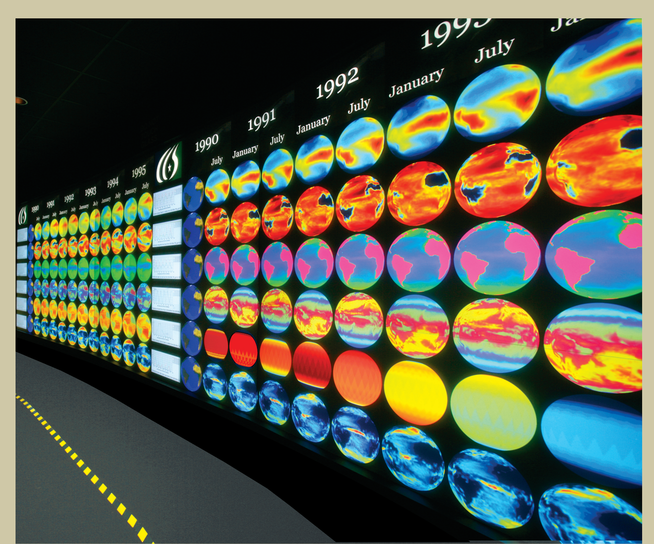
End-to-End Computing using Functional Partitioning: A Community Earth System Model (CESM) Case Study

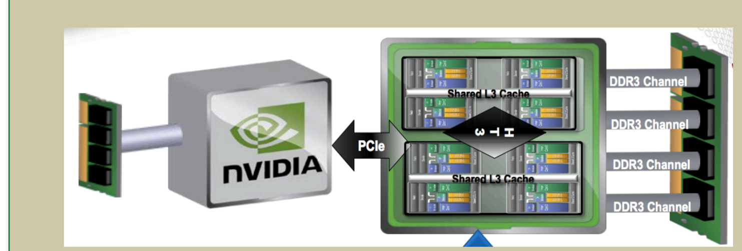
U.S. DEPARTMENT OF ENERGY

Karan Sapra, Saurabh Gupta, Ross Miller, Valentine Anantharaj, Scott Atchley, Sudharshan S. Vazhkudai, Devesh Tiwari (ORNL), Melissa C. Smith (Clemson University)

Motivation







Titan Node Structure

16 AMD Bulldozer Cores (with 8 FPUs) + 1 K20 GPU + 32 GB DRAM

- GPU is the main compute engine, hence the CPU on a node acts only as a master
- Just a few CPU cores can generate enough work to keep the GPUs busy, rest stay idle
- Underutilization of CPU resources
- Post-processing tasks such as data validation, data analytics and mining, feature extraction, and visualization pre-processing can utilize these underutilized node-local CPU resources

Our Approach

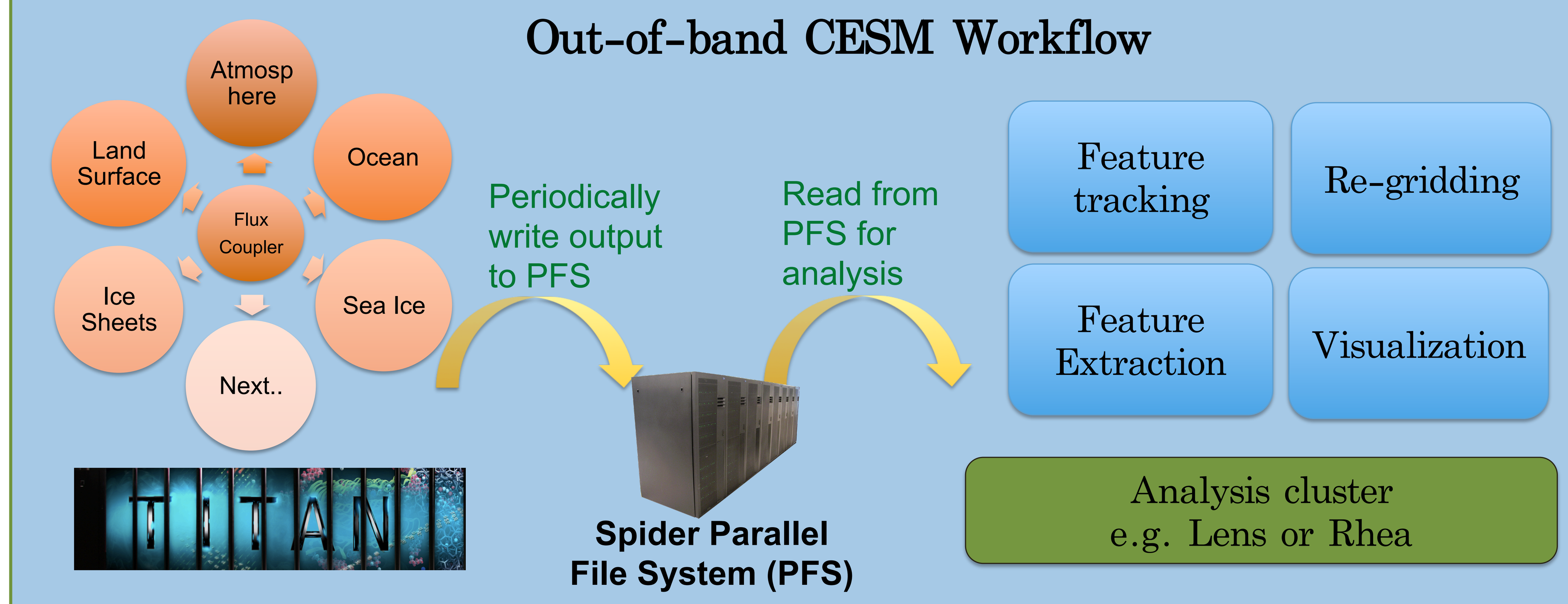
- **Functional partitioning (FP)**: A runtime framework to facilitate the exploitation of underutilized CPU cores towards an applications own end-to-end tasks
- Generic framework to express the relationship between the main task and several sub-tasks
- Simple library calls enable easier application integration and adoption

Advantages Over Existing Techniques

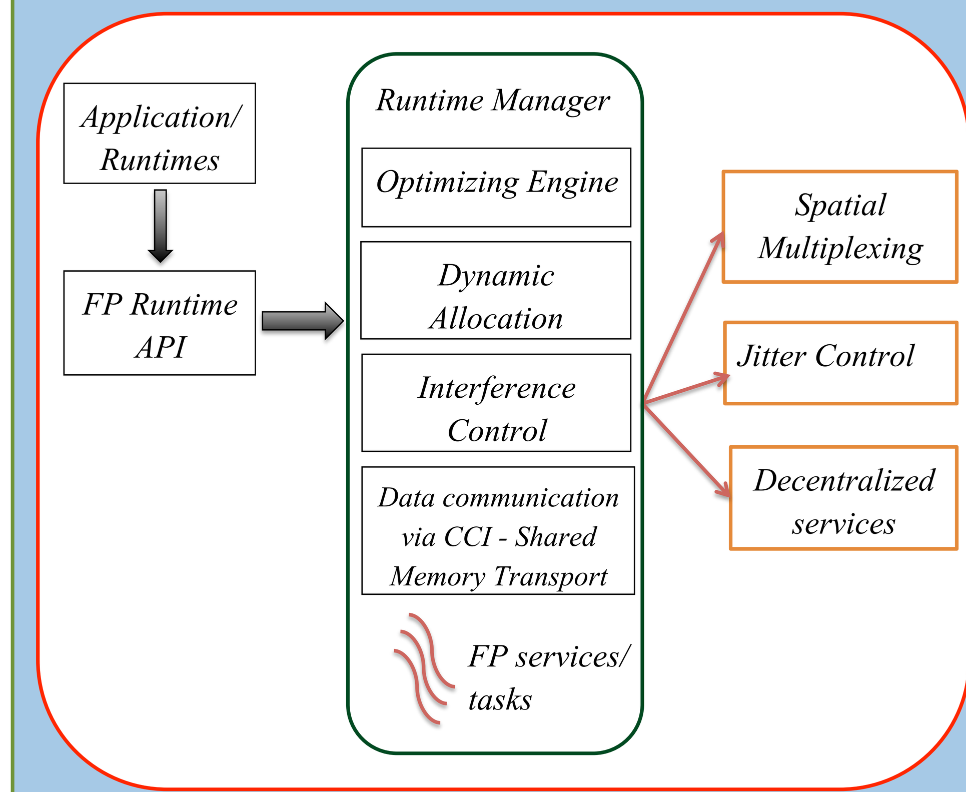
- Compute node as a self-contained entity as opposed to reserving extra nodes or an analysis cluster
- Spatially co-locating the simulation and post-processing avoids the latency in out-of-band analytics, reduces overall data movement and time to solution
- Higher resource utilization leads to better energy efficiency

Functional Partitioning (FP) Framework

Out-of-band CESM Workflow



Schematic of FP runtime



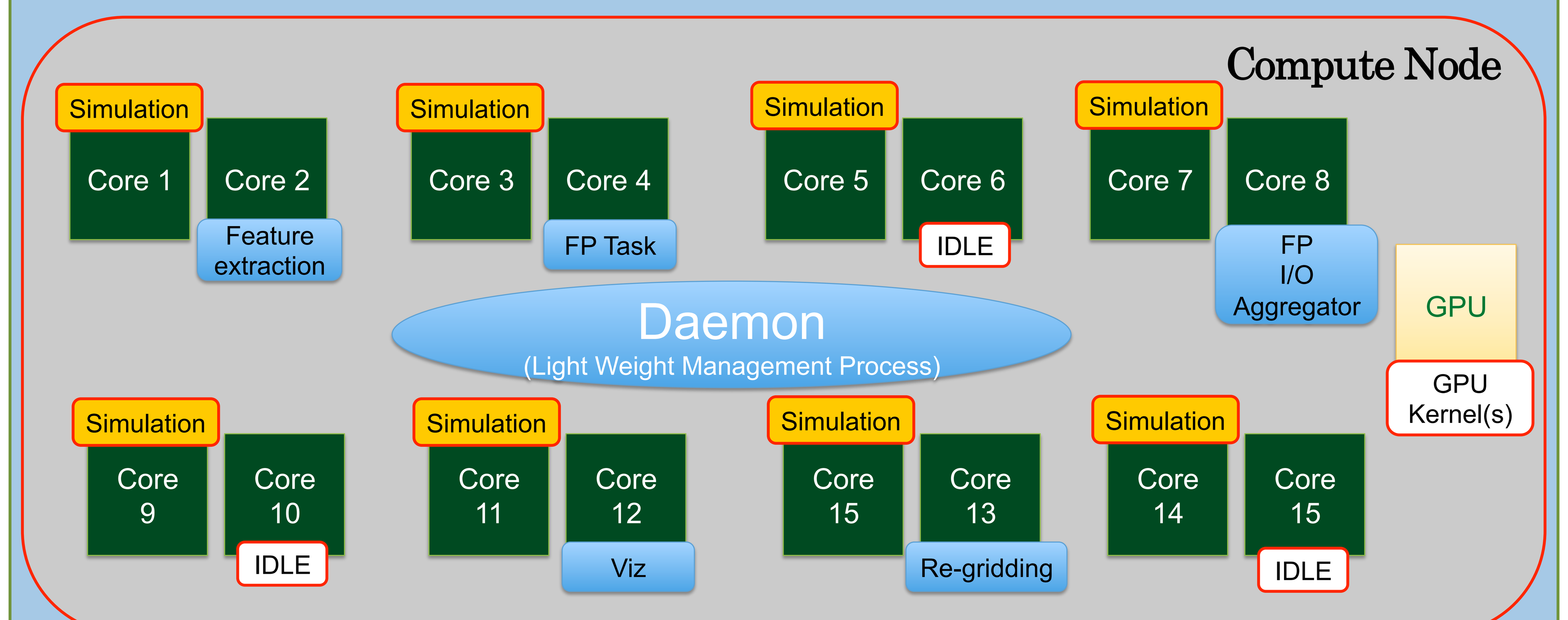
Usage Model

```
FP_Init() //Start the FP deamon
;CESM Code
..
;CESM Code

FP_Post(<args>) //FP deamon
                 creates Pthread
;CESM Code
..
;CESM Code

FP_Finalize() //Finish, Cleanup
```

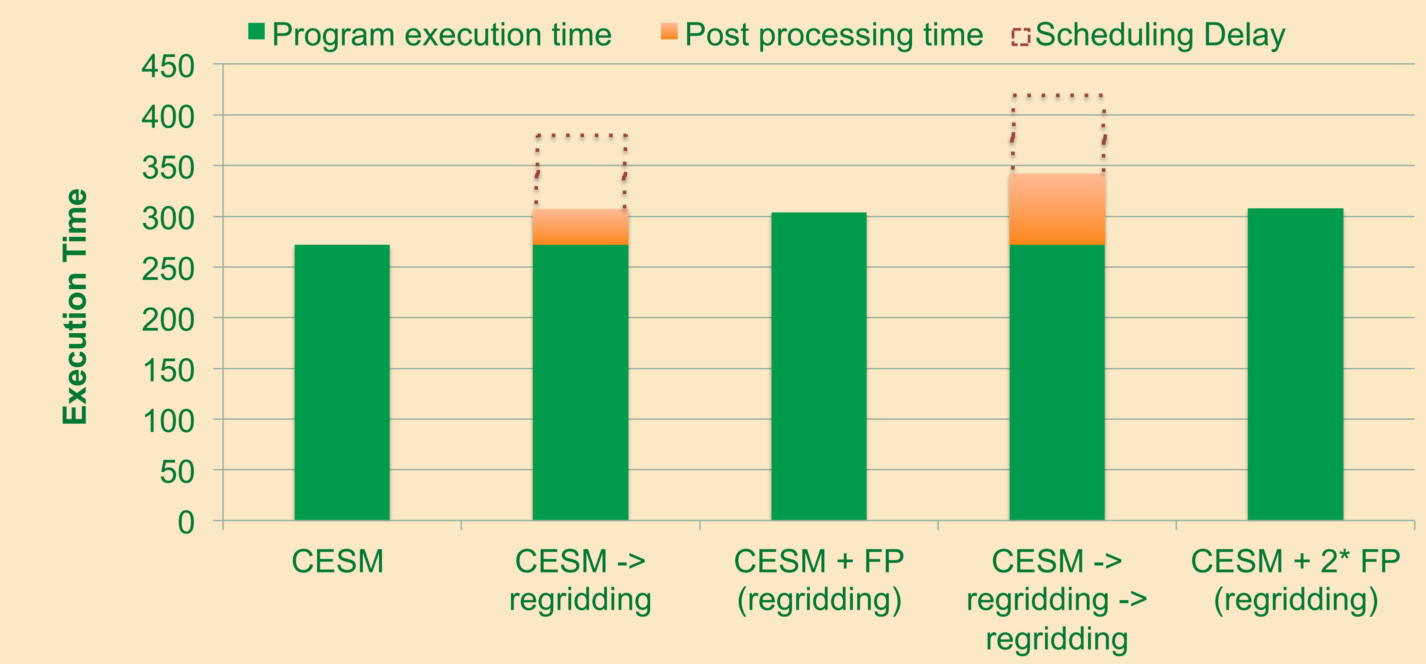
CESM Workflow with FP



FP Integration with CESM

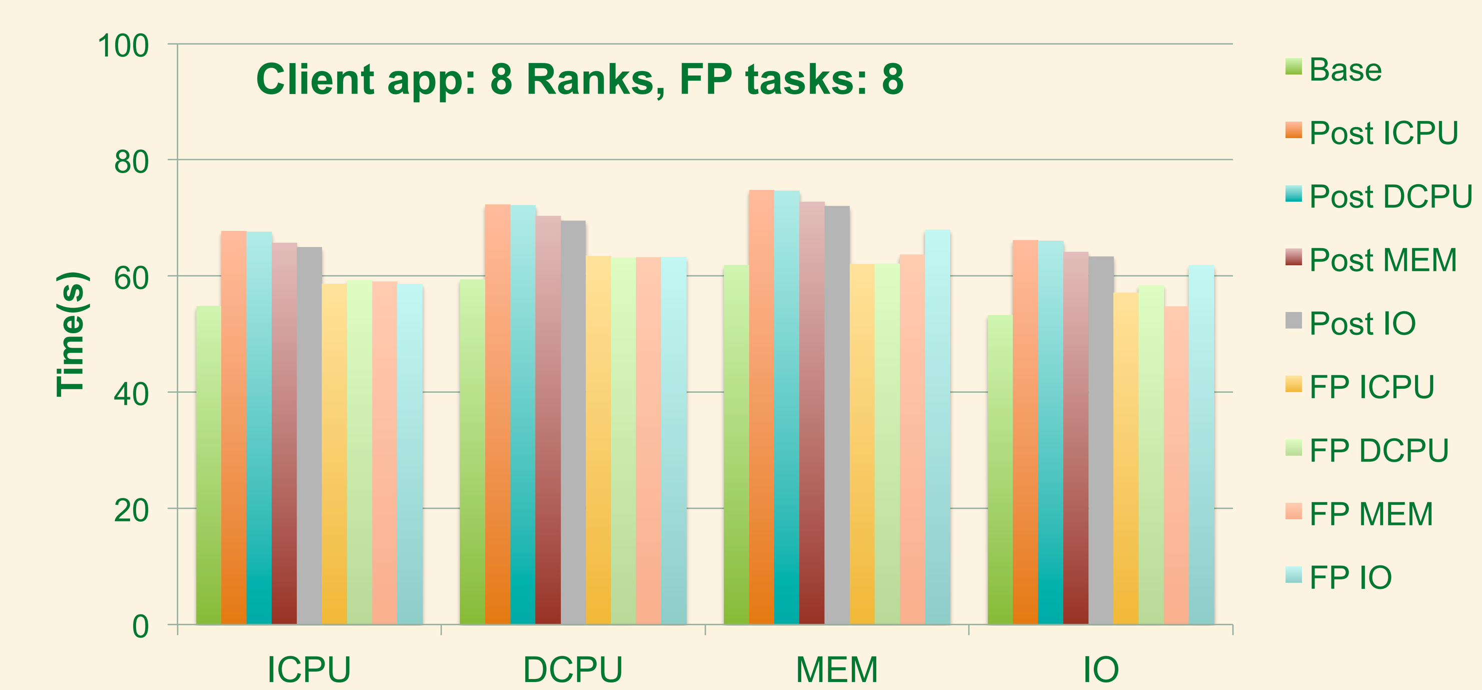
- Specify FP tasks in a configuration file
- FP tasks (Pthreads) are loaded dynamically at runtime
- fp_post() to invoke tasks after writing analysis output
- Minimal code modifications required for integration

CESM-FP Results



- FP is able to increase resource utilization and reduce the overall execution time of the CESM workflow
- Improves time-to-solution

What about Jitter?



Execution time of different types of client application (Base), main application followed by post processing (Post), and main application using FP (FP)

- Performance jitter study using micro-benchmarks that significantly stress different resources on the compute node
- Consistently reduced execution time observed when using FP compared to out-of-band post processing
- CESM is a memory intensive application and hence compute intensive workflow tasks from CESM are good candidate for end-to-end computing using FP

Acknowledgement: This work used the resources of the Oak Ridge Leadership Computing Facility, located in the National Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the Department of Energy under Contract DE-AC05-00OR22725.